back issue

TS Bulletin

for Low Cost Computing
#5-88

BILL HARMER
97 Ruskin St.,
Ottawa, Can. K1Y 4B3

This is the 16th issue in our series of newsletters devoted to the support of computer learning and experimentation using low cost computers and giving particular details of ZX-81/TS1000 news and supplier notes. The issue before you continues our series on Famous Computers with an article with the title,"When 'Home Computer' Meant Home-Made Computer". This harkens back to the days when if you wanted an affordable computer for home use you would have to build it yourself using perhaps one of several of the kits that were then available. A review of the RUN GENERATOR program currently available only for the ZX-81/TS1000 but possibly adaptable to most home computers with 16K of memory says it offers a way of getting old-fashioned style 'computer runs' from your home computer without having to program it. Insertion of a formula or two and answering a few questions will give you a neat print out or series of screen displays of interest tables or mathematical function values for example. Our ZX Hardware Hacker column continues, this time with an article on how to get a regulated 9 volts for your projects without having to turn all the parts suppliers upside down looking for the rare if non-existent 9 volt i.c. voltage regulator, the 7809. A programming feature in this issue starts what should become a series on better programming in BASIC. This one is called "On the Trail of the Spaghetti Code Monster" and should help you if you find that your programs are getting too long and complicated to debug. And our feature on Meetings, News and Supplier Notes combines the latest rumours with tidbits of information culled from a whole range of out-of-town newsletters. Find out what is going on here first. A supplement on a nearly universal BASIC dialect is also included with some issues of this Bulletin. A lament for the once touted QL is another of the articles that you may find interesting. All in all a good hunk of summer reading for the computer buff.

CONTENTS:

-ZX Hardware Hacker - The Elusive 9 volt Voltage Regulator .p.6
-Hot on the Trail of the Spaghetti Code Monster p.5
-Differing Dialects of BASIC -H.C.N.U. BASIC p.8
-Lament for the Sinclair QL p.3
-Run Generator - A Programming Tool for the Non-Programmer .p.2
-Famous Computers-When 'Home Computer' Meant Home-Made . . .p.4
-Meetings, News, Tips and Supplier Notes p.7

This is a public domain newsletter published in order to encourage the idea that you don't have to be well off to enjoy the home computing hobby. Those responsible make no guaranty of any of the technical information's correctness or the reputation of or up-to-dateness of any information on suppliers and user groups. Technical information of a hardware nature should only be used after independent verification and by someone technically qualified by formal study and diplomas as well as experience in the exact technical field. Back issues of all our newsletters, program tapes (ZX-81/TS1000 only), and booklets (such as the documentation of the Artic Assembler for the ZX-81) are all available on request and for a nominal fee to cover photocopy and mailing costs. Previous issues run about $1.50 each, the ZX-81/TS1000 tape is $7 and the booklets run about $4, postpaid.

RUN GENERATOR -A PROGRAMMING TOOL FOR THE NON-PROGRAMMER

While most of the low-cost home computers seem to be used primarily for games with some users experimenting with programming in BASIC on the side, these simple computers are fully capable of doing most of the data processing and scientific calculation tasks of the old commercial computers of the 1960's that cost as much as a million dollars.

A program has been developed (so far running only on the Timex/Sinclair ZX-81/TS1000) that will enable the non-programmer to design, by answering a few simple questions posed by the program while it is running, listings that can be printed out, of mathematical, interest or other tables, for page after page, and without having to do any programming in BASIC or any other language. All one needs is a formula (designed to run on this program) and many are available from the documentation that is supplied by the programmer of the original software.

The program is called "Run Generator I" and was written in BASIC and thus could be transferred without too much problem to other computers than the Timex/Sinclair ZX-81/TS1000/TS1500 that it was originally written for. It is designed to avoid the problem of having to program the computer for each type of special, numerical run that one might want. So it opens the power of the home computer to the non-programmer.

What kinds of run can be produced? Well, log tables and other mathematical tables can be prepared, interest schedules, depreciation schedules, tables predicting the effect of inflation and many other types of tables too. A few of the many, many possible formulas have been supplied and it would be possible to have users swap formulas and thus have the library of applications grow and grow. Any of the mathematical functions or operations allowed by the computer (in this case the ZX-81/TS1000) can be used in the formulas; logarithmic, trigonometric, exponential and even random numbers and metric conversion factors, rounded numbers and different number systems are all different possibilities.

How does the program work? The program uses a menu to start with and then the program itself asks the user to supply such particulars as how many pages to run before stopping and what title is to appear at the top of each page. The program itself takes care of the technical details of page layout and produces clean, nicely formatted print-out or screen displays as the final product.

If this was so simple, why wasn't it thought of before? In fact a highly successful computer language based on a similar idea called RPG (Report Program Generator) has been used for years by commercial data centres and computer units using IBM minicomputers as well as mainframes in the IBM 370 line. However with RPG programming was necessary and due to the more complex nature of the possible reports, that meant professional programmers.

So far however, little use of this program has been made and the writer has seen hardly any sent out to users even though it was written in December of 1986 and demonstrated at the local Ottawa-Hull Timex-Sinclair User Group around then. As far as is known no listing of it is available, but a cassette version for the TS1000/ZX-81 is available for about $7 (to cover the cost of mailing the cassette), through the publisher of this newsletter. Further, and more extended versions that were planned were never completed due to the general lack of interest.

At any rate, it seemed that this might interest those seeking such a program and so this redone version of the original article that appeared in the January, 1987 edition of the Bulletin (Bulletin #1-87) has been put in the current newsletter. Perhaps in the future other versions of this program may be prepared including a TS2068 version done through uploading and a PC compatible version in interpreted BASIC. It would depend on the interest expressed.

Lament for the Sinclair QL

With the news of A Plus one of the big suppliers for the QL and its products in the U.S. selling its inventory of QL's and realizing that the Canadian supplier Gladstone has sort of disappeared into oblivion from the Sinclair scene it would seem that the QL has passed its zenith in America.

The writer has never used either a QL or TS2068 and so is neither particularly biased for or against the computers that followed the ZX-81 and TS1000. But perhaps now it is time to look at what must have gone wrong with the QL in North America. In Europe and the United Kingdom apparently the computer is still thriving in spite of a rocky start on the home market in Britain when the computer was released with serious bugs in the ROM routines (since corrected). Over here, the main market was found among those already familiar with Sinclair product suppliers and who wanted to move up from the ZX-81/TS1000 or the TS2068. It never really had a widespread market beyond previous Sinclair product users the writer suspects. It certainly didn't penetrate the market here in the way the PC clone, Atari ST, or TI-99 did.

It was not for want of sophistocated software in the way of compilers for half a dozen languages from 'C' through Pascal to Prolog and others that were available almost from the outset. But no big name operating systems—Unix-clones, MS DOS or CP/M were ever implemented on it perhaps because of the 68000 family microprocessor which would have required an extensive simulation facility for processing in the 8086 or 8088 mode of MS DOS or the Z-80/8080 mode of CP/M. At any rate its Super BASIC was very sophisticated and programs could be used that were written for the ZX-81 or TS2068 if you wanted to key them in all over again from a listing.

The microdrive was possibly the biggest handicap in the U.S. where the 5¼ inch diskette reigned supreme at the time as a storage medium. A planned disk drive never materialized due to the financial difficulties of Sinclair and a general loss of interest by the company in the machine and the 68000 family of microprocessors.* Sinclair has gone back to the Z-80 for the Z-88 laptop computer for example and Sir Clive himself has said in an interview that moving to the 68008 microprocessor did not really accomplish much if anything that the Z-80 couldn't.

The concept of the 68008 seemed sound technically to me. Like the PC clone's 8088, it has an 8-bit bus. It has even more address lines than the 8088. And it has 32-bit rather than just the 16-bit wide registers of the 8088. However, choosing a microprocessor that no one else does, does put a machine in more risk of orphanhood than not. No other computer uses the 68008 as far as the writer is aware just as the orphanned Tandy 2000 used the equally out-of-step choice of the 80186 microprocessor. But all or nearly all of the software written for that microprocessor would be original with only the software of the 68000 with its 16-bit wide bus being at all comparable and not very much so at that. (The Atari ST, Commodore Amiga, and Apple Macintosh all use the 68000, as do some industrial 'workstations' in the high price range).

The QL has been noted for some peculiarities as far as hardware is concerned. Some have intermittent shut-down problems right out of the box which are just about impossible to fix. The computer does not come with an on/off switch. And the microdrives are tempermental and can easily be destroyed through operator error—and they are getting harder and harder to buy even at prices higher than the competition's 5¼ inch diskette.

All in all the choice of a QL has left many owners with much less hardware, software and general supply (microdrive replacement) support than other brand loyalists who are not so heavily handicapped as well as of course those who own the ubiquitous PC clones. That said, in a way, the writer still admits to a desire to own a QL. And the users are not just going to go away as many local user groups and some suppliers do still support them.

---

*Other vendors have since produced disk drive systems for the QL.

FAMOUS COMPUTERS: When'Home Computer'Meant Home-Made Computer

Once upon a time, getting a home computer meant building one from a kit or plans in a magazine. One of the earliestthat the writer can remember was a sort of mechanical beast advertised in the 1950's in home mechanics' magazines and which from the pictures looked like an erector set with three big wheels with many holes punched in them as the working parts. Real computers at the time used vacuum tubes, had a few K of memory and cost about $1 million at a time when that was worth about ten times what it would amount to in purchasing power today.

Heathkit, which sold hi-fi's in kit form, and oscilloscopes (using originally war surplus radar CRT's), was one of the early computer kit retailers. They had an analogue computer (for lab use) that cost about as much as a popular sub-compact car. Later they sold a clone of the DEC PDP-8 computer in kit form. But kit computers really came of age with the marketing of the 8008 microprocessor (on a silicon chip), forerunner of the 8080 (predecessor of the Z-80). The 8008 was the first popular 8-bit computer microprocessor that caught the imagination of the hobby market and both the Heathkit H-8 and the Altair were based on this family of microprocessor.

In those days a computer had no screen or keyboard so that a separate dumb video terminal perhaps bought on the surplus market or a teletype terminal bought surplus was used for input and output. The first home computers used an integer BASIC interpreter (no decimal fractions allowed) available on punched paper tape to go with the teletype machine input. Only later did the use of ordinary cassette tapes come into use based on a standard determined at a meeting in Kansas City.

A book published in 1979 ("How to Design, Build and Program Your Own Working Computer System" by R.P. Haviland, TAB Books) was one of the early references in this mode. It featured the Motorola SCAMP 8-bit microprocessor and so was not as popular as a book using the 8008 or 8080 would have been. It is a treasure trove of practical information from the design and construction of the metal cabinet to house the computer to the design of an operating system. It includes instructions on how to build your own ROM using an array of silicon diodes. Computer building was however a big step up from building your own "Pong", tennis type video games that were all the rage in 1972 and for which plans appeared in many electronic magazines.

The real granddaddy of home computers was the MITS Altair 8800 using the 8080 microprocessor and sold mailorder in kit form from 1975 on. The design was copied by other manufacturers and the S-100 bus (100 for 100 lines) was established as an early standard for connecting peripherals and accessory cards and expansions. The Altair was featured on the cover of the January, 1975 edition of Popular Electronics magazine (now defunct). The kit sold for a few hundred dollars. Out of such home computers and experience at school and work with commercial computers, meetings of a home brew computer club in Menlo Park , California (in a member's garage) brought together youngsters and engineers who would start a couple of dozen computer firms including Cromemco, North Star, Osborne and Apple.

How many people who bought these computer kits would actually finish building them and get them debugged and working is a good question. Some possibly still roost in attics or back closets or basements, with too many bad solder joints or crossed wires to to ever be put into working order. Some undoubtedly were used as practical computers until the Commodore PET, TRS-80 Model 1, and Timex/Sinclair (also sold in kit form once but much simpler to build) made home computing accessible to schools and hobbyists who did not have the patience or skills of the soldering iron jockeys of the early days. It is hard to remember how recent were those "early days".

Hot On The Trail of the Spaghetti Code Monster

Worked on that masterpiece program for weeks now, stretching into a couple of months and you still can't get it to run without strange halts and bizarre twists and turns of logic? Do you find that every time you set out to track down the problem, you understand the convoluted sequence of program flow less? Got variables stuck in there that you have long since forgotten what they are supposed to be doing if anything now after a few modifications later? Well, booby, you have got the spaghetti code blues once again—just after you promised yourself the last time that you would try to learn structured programming.

Structured programming is not something that you can't practice in home computer BASIC. Any version or dialog of BASIC will do—you just use it differently than you used to. They can all be adapted to get the most important principles of structured programming into action—and who cares about the rest? Because structured programming is a set of rules for programmers not programming languages at the bottom of it and anyone can use them or abuse them. And abuse means speghetti code for any program longer than one or two K (and even a few of the shorter programs like that can be a mass of spaghetti code, if you are really determined to try to complicate them).

Why does structured programming work and why can't you debug your own invented program structure that is a mass of spaghetti code? Well, the problem with spaghetti code is that it makes for complicated programs. The bigger the spaghetti code part of the program, the more complicated. The second principle is that a given programmer (depending on intelligence quotient improved a little by experience) can only debug a program of a given size and complexity combination. If the complicated program module grows beyond that he will never, ever without some sort of fantastic stroke of pure luck be able to debug it when it develops mysterious problems which stop it from running smoothly in some (or all) cases (of input combinations).

So the solution is modular programming. Break the program up into smaller modules, each module assigned a part of the programming task and each small enough and simple enough to be well within the range of debugging skill demanded of the programmer writing the program—that is you and me. One way of doing this is to break the program into blocks, say input, calculation and output at the largest level and move between these blocks by calling them up by GOSUB or GOTO (in some computers that have limited GOSUB capabilities) instructions in a further, main or control block of the program. Ideally for logic this main block would be say from lines 1-500 but to speed up a program in already slow ROM'ed BASIC, it is better to put it at say lines 8000-8999 so that the calculation or other parts of the programs using much looping can be placed in the faster running 1-1000 line area. (Most interpreted, that is ROM, and a few other disk or cartridge BASIC's run loops faster when they are placed in low numbered lines with no other statments ahead of them. Exception: Sinclair QL SuperBASIC. That is because in restarting a loop, the computer flips through all the line numbers from one on to find where to begin again).

And other complicated tasks especially in the calculation part of the program can similarly be broken down further into smaller modules and each one can be tested and debugged separately as if a complete program of only that size and complexity. The principle or rule of what makes a module is that you must only enter it from one place and leave it from one place (by GOTO's for example) or it ceases to be a self-contained module and becomes instead just a chunk of a larger mass of spaghetti coded, programmer frustrating, after-midnight debugging time eating, non-structured chaos.

And that is the main principle of structured programming, breaking tasks or jobs for the program to do into smaller and smaller modules each with only one entry and one exit point (althought you may exit them to call in a subroutine which is a smaller module as many times as you want). (To be Continued).

[illegible]

ZX - Hardware Hacker - The Elusive 9 Volt Voltage Regulator

A most convenient way of getting a power supply of a given voltage output (or even several voltage outputs from the same transformer winding source after rectification and filtering) is to use an integrated circuit voltage regulator of the 78xx family. This family is used for obtaining voltages of 5, 8, 12, 15 and a few other higher voltages by using a pass transistor on its silicon chip to drop down the voltage of the d.c. power supply fed to it to its rated output voltage. The problem is that many circuits and especially the type that hobbyists build are designed to work from 9 volt batteries and to make a power supply for them using one of these voltage regulators you soon find that 9 volt voltage regulators are hard or impossible to find. On looking through a local parts store for example the writer could find a 78L09 rated for very Low current output but no 7809 (the standard one rated for about 1 amp. maximum current output if heatsunk).

So the question is, "How do you get 9 volts out of a voltage regulator?" There are a number of possible solutions. One would be to design a voltage regulator from scratch using discrete components. Plans could be found perhaps in a book of voltage regulator circuits or a computer program to design one could be found (one was published in Radio-Electronics, about five years ago). But that is hardly a quick and easy fix. One could build a variable voltage regulator using say an LM317 and just set it manually for 9 volt output—but if the setting gets off and you deliver an overvoltage to your equipment, as might easily happen, kiss your equipment goodbye! It could be fried into a non-functional state in a matter of seconds.

The other thing is that if you can get an 8 volt voltage regulator that might be quite good enough to run equipment designed to run from a 9 volt battery. It often will run on lower voltages like the 8 volts provided by a 7808 since the designs often allow for running it from Ni-Cad batteries that average 7.2 volts output (but have less voltage fall off than standard batteries under load). So the regulated (and thus less subject to fall off) voltage of a 7808 at about 8 volts may be quite adequate.

But if you really want to hit 9 volts or there abouts you can fool a 7808 into thinking that it is sensing only 8 volts accross its output even though it is outputing a higher voltage by inserting forward biased diodes in its sense line (the middle pin). If you assume that a typical silicon PN diode like the 1N4002 has a drop of about 0.6 to 0.9 volts accross it when forward biased, and a Schottky diode like the 1N5822 about half that, you can increase the output voltage in steps of these voltages by inserting them (cathode band pointing towards ground) in the sense line. This reduces the stability of the output voltage maintenance slightly and diodes vary as to their voltage drop so that some experimentation would be necessary with a meter placed on the output before actually using the device but it would seem theoretically that this method of design and experimentation in the hands of a technician or experienced and knowledgeable hobbyist would fill the bill. At the very least it would save on gas and long distance calls as one futilely combs all the suppliers looking for the all but extinct 7809 voltage regulator i.c.

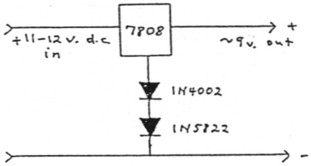

Fig. 1 - 9 volt Approx. V.R.
(See text)

Meetings, News, Tips and Supplier Notes

E.A. Brown is selling off its Sinclair books and products by the case lot. This offer was made in April. Phone (612) 762-8847 if you are a dealer or user group that wishes to bid on any remaining lots....Indiana User Gp. editor Mike Felerski reports on sending his Spectrum 128 back to England (to Video Vault) for service and being pleased with the results....QZX a magazine for the use of Sinclair Computers (ZX-81/TS2068) in amateur radio is available -2025 O'Donnell Dr., Las Cruces, NM88001, USA....The missile fuel plant that exploded in Henderson, Nevada was 3 miles away from the Las Vegas Group's newsletter editor whose house received some damage in the form of cracks in the walls but no broken windows....A + Computers, a QL dealer is selling off in a lot its new QL's although will continue to provide some other products for the QL. Curry Computers and RMG Enterprises as well as Sharp's still support the QL in the US but new computers may have to be imported directly in the future at higher prices....Larken Disk Drive makes good impression in demo at Las Vegas User Gp. meeting in May according to their newsletter....Geo. Chambers of the Toronto TS User Gp. offers this short program to erase a number of files on the Larken TS2068 disk system:

```
10 RANDOMIZE USR 100: CAT"",
20 INPUT a$
100 RANDOMIZE USR 100: ERASE a$, : GO TO 20
```

(from Sinc-Link
(May-June 88 Newsletter
...Thanks George.

....Another tip for TS2068 programming from John Van der Stel, Michigan in the same newsletter: use PAUSE 0 instead of IF INKEY$="" THEN___, to have the computer pause waiting input from the keyboard and another tip to enable the capital lock within a BASIC program - IF CODE I$=6 THEN POKE 23658,(8 AND PEEK 23658=0) - this is all one line and thus takes little memory....Unfortunately as was mentioned in the last issue, a previous issue of this newsletter (#3-88) repeated a report that implied that Knighted Computers was going out of Timex-Sinclair. In fact Knighted Computers, 10 Canalview Mall, Fulton, NY 13069, USA has every intention of continuing the support of our computers. A recent addition to their line is a Lottery program that analyses numbers(in the local N.Y. draw it is believed)in much the same way that a recent winner used. Unfortunately a letter the writer sent them to apologize was returned by the post office to further confound the situation.... Recent meetings at the Ottawa-Hull TS User Group have included demos of surface mount electronic components and the Disciple board for the Spectrum ROM TS2068. Attendance seems to be holding up well at about ten members per meeting. Although this newsletter has no formal connection to the user group, the newsletters sent to us in exchange for this one are generally turned over to the library of the user group....The writer has been working on and off on a program for the ZX-81 to turn some 6800 subroutines into Z-80 assembly language. It has not proven as simple as just making up a table to convert assembly language instructions one to one. In addition there are memory problems with the program wanting to take up more than 16K even if only a subset of the 6800 instruction set is allowed. All in all it is a project more educational than practical.... A list of special interest BBS numbers in the Las Vegas Hacker n/L included Inventors & authors at 817-430-3760, Hare Krishna at 312-743-7729 (USA), World Peace at 303-320-4822, Handicapped at 518-793-9574 and Crypto at 703-237-4322 hrs.18:30-7:30 EST weekdays and 24hrs. wk. ends.... At the June Ottawa-Hull TSUG meeting member David Solly did a demo on using the Hisoft Pascal compiler for the TS2068 with the Larken Disk (cartridge port) system and showed how a couple of machine code routines within a Pascal program can be used to save data (such as database contents) to disk....Ken Schieman, 18 Columbus, Ottawa, Can.K1K 1R3 - (613)749-6277 is looking for FORTH compiler to run on his ZX-81/TS1000 other than Artic's ZX-FORTH. If anyone has a PluriFORTH cartridge (by Tree Software) or other info on FORTH compilers he would appreciate hearing from you....S. Petrovic, 1007 Joseph St., Apt.2, Lafayette, IN 47905, USA has a mailing list program for the TS1000-ask for info

Differing Dialects of BASIC

Wouldn't it be nice if programs written in BASIC for any home computer would run on them all? One might think that the only problem is the stubborn insistence of the different home computer manufacturers of a few years ago when our favourite machines were designed, in including features in one BASIC dialect that were not available or were formatted differently in another. But that is not the only problem. You are forgetting about memory!

A computer program designed to be 'lean and mean' and perhaps having as a result crummy or even no graphics due to being shoehorned into a RAM requirement of one, two or three K (as with the Sinclair ZX-81, TS1000 and Radio Shack MC-10 and VIC-20 respectively) may be made to run on a bigger computer with much more memory but it will look sick beside programs designed to take advantage of its larger capacity. And of course there is no way other than a total rewriting from scratch to fit a big program into a little computer in most cases, if it is possible at all. Oh yes, you can separate it into modules of different functions, but if you want to use all or several of the functions on one set of data keyed in, you will have to load each module separately and key in the data all over again. Few of the really small computers (and possibly none as far as the writer knows) have a provision to load data by say cassette separate from a program.

So what you have is a series of levels of sophistication in programs based on memory requirements, with the one, two and three K models at the bottom, the 16K models in the middle (many of the small ones were expandable to at least 16K), and most of the early home computers limited to about 32K of BASIC program lines (even though they were often advertised as 64K machines, including the Commodore 64, ZX-81/TS1000 and TS2068 and others). At the real top are the PC clones, and 68000 based machines running compiled and professionally written programs taking up more than 256K and often by the use of loading from a floppy disk or two, as required, having the whole disk to fill with programs or program options.

So any attempt to standardize BASIC programs would have been doomed from the start by the largely unforeseen drop in memory prices that outran even the most optimistic predictions. But we could still run simple programs to do simple things in say all computers that had a minimum of memory if we could find a subset of instructions that exist in virtually all dialects of BASIC. It is exactly this that the writer is proposing to do. We would have to forego the use of PRINT AT, (for in some computers it is PRINT @ plus one number instead of two and in some replaced by the separate command LOCATE which does not call for printing at all), LEFT$ (which in Atari's and ZX-81/TS1000 lines not used), INKEY$ and GET, and a few other (actually a lot of other) commands and whittle the list down to a dialect of Home Computer Nearly Universal BASIC that would get simple computing jobs done on nearly any computer! A BASIC which would be a subset of nearly all BASIC dialects would answer our needs.

Of course as far as graphics are concerned there could be virtually no special graphic characters unless the character set of any computer could be guaranteed to have it. This would mean limiting the character set to A to Z, Ø to 9 and some punctuation and arithmetic characters. But even this would allow us a little in the way of graphics, if you think about it, including a line of periods or minus signs for example to make a frame.

There would also be an advantage to the limitations on program size in that since these programs would be circulated on paper (due to different cassette standards etc.) and having to be keyed in by hand, keeping them short due to memory limitations would also keep them shorter in entering them by the keyboard.

So programs written in this dialect would be short and functional, no fancy graphics or mouse-driven option selection, "Just the facts ma'am, nothing but the facts..." as the old t.v. program used to say. And that would turn many a computer including orphaned ones into useful tools for everyone.